Project Review

Meets Specifications

Awesome work 🎉 You have put in great efforts to work through the entire project.

This is a very well done project! Your code is clean as well. It's evident that you have picked up AWS Sagemaker and got yourself familiar with most of the concepts and were able to implement the project with great understanding. Kudos on that 👏

Congratulations once again 🎉 Happy Learning and wishing you the best for all your further learning and playing with neural networks as this is your final project for graduation 😉

Leave a rating if the review was helpful 😇

Files Submitted

The submission includes all required files, including notebook, python scripts, and html files.

Make sure your submission contains:

- **The `SageMaker Project.ipynb` file with fully functional code, all code cells executed and displaying output, and all questions answered.**
- **An HTML or PDF export of the project notebook with the name `report.html` or `report.pdf`.**
- **The `train` folder with all provided files and the completed `train.py`.**
- **The `serve` folder with all provided files and the completed `predict.py`.**
- **The `website` folder with the edited `index.html` file.**

All the required files are submitted 👍

Preparing and Processing Data

Answer describes what the pre-processing method does to a review.
Nice that you pointed out it removes the stop words. Also we use `html` parser to remove the `html` tags from the sentences to make sure the tags which are of no significant importance don't contribute in deciding the sentiment of any word

Tips

- Further reading on tokenization https://nlp.stanford.edu/IR-book/html/htmledition/tokenization-1.html

The `build_dict` method is implemented and constructs a valid word dictionary.
Really enjoyed going through your implementation. We can use the `Counter` method from the `collections` module for this to reduce the amount of code needed.

Easy to read and a lean Implementation 👍

Counter usage

```
word_count = Counter(np.concatenate(data, axis = -0))
```

Notebook displays the five most frequently appearing words.
Nice work and also it clearly makes sense that `movi, film, one, like, time` are the words cause it's a movie dataset ✅
Answer describes how the processing methods are applied to the training and test data sets and what, if any, issues there may be.
Also we should note that extra care has to be taken during processing the data so we don't introduce any data leakages:

- Since the `preprocess_data` is applied on `per/record` basis there's no problem there.
- Also if you notice `convert_and_pad_data` doesn't cause an issue as the `word_dict` is only constructed from training data set.

Build and Train the PyTorch Model

The train method is implemented and can be used to train the PyTorch model.
Awesome work on clearly implementing all the required steps which is Feed Forward, Calculating losses via loss function and then updating the weights via BackPropagation 👏
The RNN is trained using SageMaker's supported PyTorch functionality.
Excellent work on utilizing AWS Sagemaker to perform the training 👏

Tips

- More reading on bringing your own algorithms and running them on AWS infrastructure https://docs.aws.amazon.com/sagemaker/latest/dg/your-algorithms-training-algo.html

Deploy the Model for Testing

The trained PyTorch model is successfully deployed.
I can see that you were able to deploy the model with ease. Really nice work 👍

Use the Model for Testing

Answer describes the differences between the RNN model and the XGBoost model and how they perform on the IMDB data.

Make sure your answer includes:

- **The comparison between the two models**

- **Which model is better for sentiment analysis**

`XGBoost` https://docs.aws.amazon.com/sagemaker/latest/dg/xgboost.html is provided by aws out-of-the-box.In our case we have used our own algorithm and our own containers to do the training which shows the flexibility of sagemaker as a platform to bring our own algorithms and perform training.

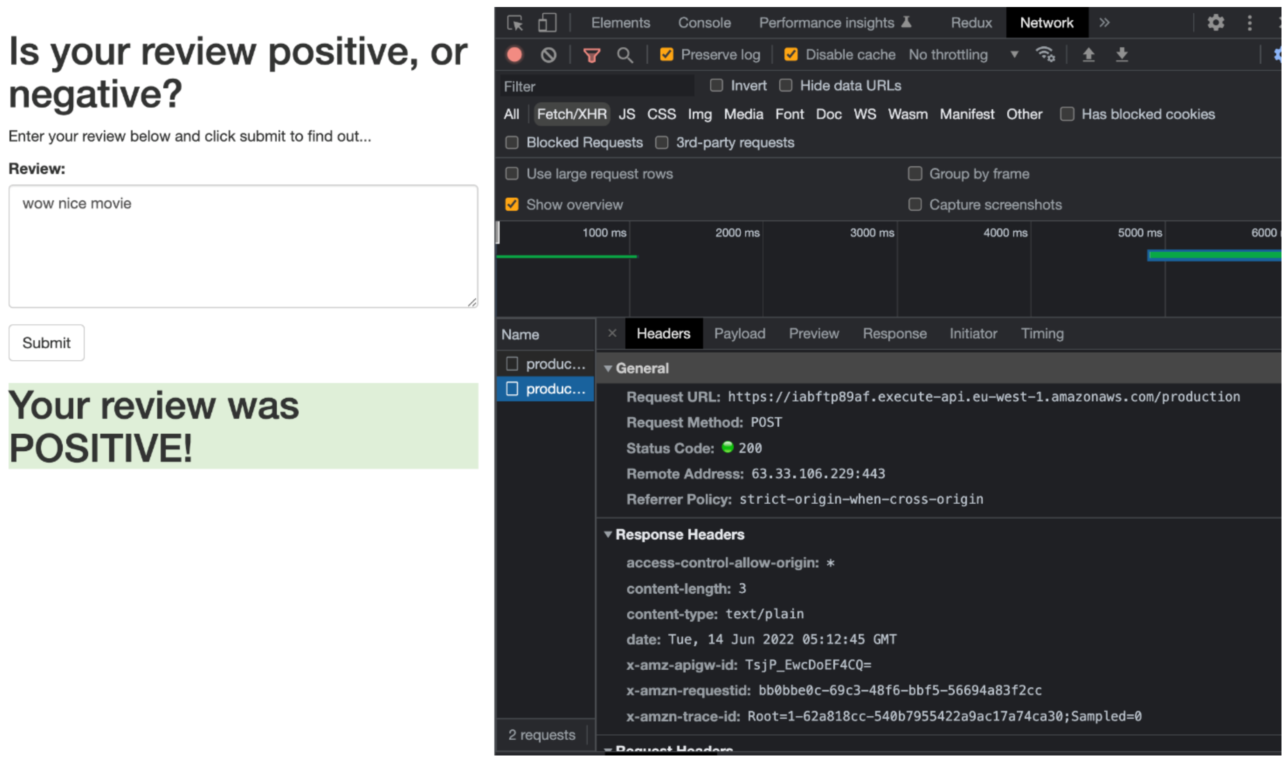
Our model might not out perform XGBoost but we now know the fundamentals of sagemaker..
The test review has been processed correctly and stored in the `test_data` variable. The test_data should contain two variables: review_len and review[500].
Good work on utilizing `review_to_words` and `convert_and_pad` functions to preprocess our review and then calling the predictor on it.
Nice work 👏
The `predict_fn()` method in `serve/predict.py` has been implemented.

- **The predict script should include both the data processing and the prediction.**
- **The processing should produce two variables: data_X and data_len.**

The `predict_fn()` implementation is clean. I like your method since you are packing the data using `np.hstack` and also utilizing the gpu available by creating tensor representation using `torch.from_numpy`

Deploying the Web App

The model is deployed and the Lambda / API Gateway integration is complete so that the web app works (make sure to include your modified `index.html`).
I can see that your have deployed your model and have utilized the endpoint in the web page 👍


Is your review positive, or negative?

Enter your review below and click submit to find out...

Review:

wow nice movie

Your review was POSITIVE!

```
▾ General
    Request URL: https://iabftp89af.execute-api.eu-west-1.amazonaws.com/production
    Request Method: POST
    Status Code: ● 200
    Remote Address: 63.33.106.229:443
    Referrer Policy: strict-origin-when-cross-origin
▾ Response Headers
    access-control-allow-origin: *
    content-length: 3
    content-type: text/plain
    date: Tue, 14 Jun 2022 05:12:45 GMT
    x-amz-apigw-id: TsjP_EwcDoEF4CQ=
    x-amzn-requestid: bb0bbe0c-69c3-48f6-bbf5-56694a83f2cc
    x-amzn-trace-id: Root=1-62a818cc-540b7955422a9ac17a74ca30;Sampled=0
```

The answer includes a screenshot showing a sample review and the prediction.

Good basic samples supplied and tested .

I have played around with the website by providing few samples and the endpoint was returning the correct sentiment. Kudos on the work deployed.